M2A – Sorbonne Université

Reinforcement Learning Project 2

Félix Bos, Xu Shuo

October 2025

1 Introduction

Reinforcement Learning (RL) has achieved remarkable success in solving complex sequential decision-making problems. However, most RL algorithms are based on the assumption of a fully observable Markov Decision Process (MDP), where the current observation contains all information necessary to determine the optimal action. In many real-world scenarios, this assumption does not hold due to sensor noise, occlusion, or missing measurements, leading to a setting better described as a **Partially Observable Markov Decision Process (POMDPs)**. Under such partial observability, standard RL agents struggle to infer hidden dynamics, often resulting in unstable learning and slower convergence.

To mitigate this limitation, this study introduces a lightweight temporal structure enhancement method called **Temporal Extension**. Instead of relying on recurrent neural networks, it explicitly incorporates temporal dependencies at the input and output levels. Specifically, *observation stacking* provides short-term memory by combining several consecutive observations, enabling the agent to infer hidden dynamic variables such as velocities. Meanwhile, *action chunking* allows the agent to predict a short sequence of future actions, improving control smoothness and robustness in continuous tasks.

The proposed approach builds upon the **Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm**, which stabilizes training in continuous control by employing twin critic networks, delayed policy updates, and target policy smoothing. In this work, TD3 serves as the baseline framework for integrating the temporal extension mechanism, enabling a systematic evaluation of its effectiveness under various levels of observability.

The study focuses on two main research questions:

1. How does partial observability, caused by hiding critical state variables such as horizontal or angular velocity, affect the performance and convergence of the TD3 algorithm?

2. To what extent can temporal extension mechanisms (combining observation stacking and action chunking) restore or stabilize TD3's performance under partial observability?

2 Methodology

2.1 Background POMDP

This study focuses on applying reinforcement learning (RL) to a **partially observable continuous control task**. We adopt the `LunarLanderContinuous-v3` environment, where the agent (the lunar lander) must control the thrust of its engines to achieve a smooth and safe landing on a designated platform under limited fuel conditions. At each time step t, the agent receives an observation o_t from the environment, selects an action a_t according to its policy $\pi(a_t|o_t)$, and obtains a reward r_t based on the resulting state transition.

State and Observation. The true state of the lander $s_t \in \mathcal{S}$ consists of eight continuous variables:

$$s_t = [x,\, y,\, v_x,\, v_y,\, \theta,\, \dot{\theta},\, c_1,\, c_2],$$

where:

- x, y: horizontal and vertical positions of the lander,
- v_x, v_y: horizontal and vertical velocities,

- θ: orientation angle,
- $\dot{\theta}$: angular velocity,
- c_1, c_2: binary contact indicators for the two landing legs.

In the **fully observable** environment, the agent has access to all eight variables. To simulate partial observability, we manually mask certain velocity components, constructing the following observation variants:

$$\begin{aligned} o_t^{(\text{No\_Vx})} &= [x,\, y,\, v_y,\, \theta,\, \dot{\theta},\, c_1,\, c_2], \\ o_t^{(\text{No\_Vtheta})} &= [x,\, y,\, v_x,\, v_y,\, \theta,\, c_1,\, c_2], \\ o_t^{(\text{No\_Both})} &= [x,\, y,\, v_y,\, \theta,\, c_1,\, c_2]. \end{aligned}$$

Under these partially observable conditions, the agent no longer has direct access to full dynamic information, especially velocity-related features, which may lead to unstable training and degraded performance.

Action and Reward. The continuous action space is two-dimensional:

$$a_t = [a_{\text{main}}, a_{\text{side}}],$$

where:

- a_{main}: main engine thrust, controlling vertical movement,
- a_{side}: side thruster force, controlling horizontal motion and orientation.

The environment provides a shaped reward signal that considers several aspects:

- positional and angular deviation from the landing pad,
- smoothness of velocity during descent,
- fuel consumption penalty,
- bonuses for successful landing and penalties for crashes.

2.2 TD3 Algorithm

The **Twin Delayed Deep Deterministic Policy Gradient (TD3)** algorithm is an enhanced version of DDPG for continuous control tasks. It belongs to the *off-policy actor–critic* family and stabilizes training through three key techniques: (1) twin critic networks, (2) delayed policy updates, and (3) target policy smoothing.

Framework. The agent includes an actor network $\pi_\theta(s)$ producing deterministic actions $a = \pi_\theta(s)$, and two critic networks $Q_{\phi_1}(s,a)$ and $Q_{\phi_2}(s,a)$ estimating the state–action value $Q(s,a)$. TD3 improves upon DDPG by taking the minimum of the twin critics to reduce overestimation, updating the actor less frequently to stabilize learning, and adding clipped noise to target actions for smoother updates.

Network Structure and Update Frequency. TD3 maintains both online and target networks. Update frequencies are summarized below:

Component	**Role**	**Update Frequency**
Q_{ϕ_1}, Q_{ϕ_2}	Online critic networks	Every step
π_θ	Online actor network	Every d steps (delayed)
$Q_{\phi'_1}, Q_{\phi'_2}$	Target critic networks	Soft update with actor
$\pi_{\theta'}$	Target actor network	Soft update with actor

The delay step is typically $d = 2$.

Optimization Objectives.

(1) Critic Objective. The critics minimize the Bellman error:

$$\begin{aligned} a' &= \pi_{\theta'}(s') + \epsilon, \quad \epsilon \sim \text{clip}(\mathcal{N}(0,\sigma), -c, c), \\ y &= r + \gamma(1-d)\min(Q_{\phi'_1}(s',a'), Q_{\phi'_2}(s',a')), \end{aligned}$$

$$\mathcal{L}_{\text{critic}}(\phi_i) = E_{(s,a,r,s')\sim D}\Big[(Q_{\phi_i}(s,a) - y)^2\Big], \quad i = 1, 2.$$

(2) Actor Objective. The actor maximizes the critic-estimated return:

$$J_{\text{actor}}(\theta) = E_s[Q_{\phi_1}(s, \pi_\theta(s))], \quad \mathcal{L}_{\text{actor}} = -J_{\text{actor}}.$$

The actor is updated once every `policy_delay` steps.

Target Network Update. All target networks are updated via Polyak averaging:

$$\phi_i' \leftarrow \tau\phi_i + (1-\tau)\phi_i', \quad \theta' \leftarrow \tau\theta + (1-\tau)\theta', \quad \tau \approx 0.005.$$

Through these mechanisms, TD3 effectively suppresses overestimation bias, stabilizes critic learning, and enhances policy robustness in continuous control environments.

2.3 Temporal Extension Mechanisms

In partially observable environments, the current observation o_t often fails to capture the complete underlying state s_t. To alleviate the performance degradation caused by missing information, two complementary **temporal extension mechanisms** are introduced in this study: (1) a memory-based observation extension to integrate past information, and (2) a prediction-based action extension to enable short-term planning. Both mechanisms are implemented within the class `ObsTimeExtensionWrapper(gym.Wrapper)`, which modifies the environment's observation and action interfaces to extend temporal representations.

Observation Extension (Memory-based). The observation extension provides the agent with short-term memory by stacking several consecutive observations. At each time step, the agent receives a concatenation of the most recent k frames:

$$\tilde{o}_t = [o_t, o_{t-1}, \ldots, o_{t-k+1}].$$

This design allows the policy to infer hidden dynamic features such as velocity or angular rate from historical changes, thereby modeling temporal dependencies without relying on recurrent neural networks.

Action Extension (Prediction-based). The action extension enables the agent to perform short-horizon prediction and planning. In this mode, the actor network outputs a sequence of future actions:

$$[a_t, a_{t+1}, \ldots, a_{t+m-1}] = \pi_\theta(\tilde{o}_t),$$

where only the first action a_t is executed in the environment, while the remaining predicted actions serve as a short-term plan to improve control smoothness and robustness.

Combined Extension (Both). When both mechanisms are activated simultaneously, the agent benefits from short-term memory of past observations and predictive foresight of future actions. This configuration, referred to as `Both`, demonstrates the most stable and effective performance in our experiments.

3 Experiments

3.1 Experimental Setup

All experiments were conducted on the continuous control task `LunarLanderContinuous-v3`, using the TD3 algorithm described in Section 2.2. The experiments were implemented in `PyTorch` with the training environment built on `Gymnasium`. A total of two million environment steps were performed for each configuration. The random seed was fixed to ensure reproducibility.

Table 1 summarizes the main hyperparameters used in all training sessions.

Table 1: TD3 hyperparameters used in all experiments.

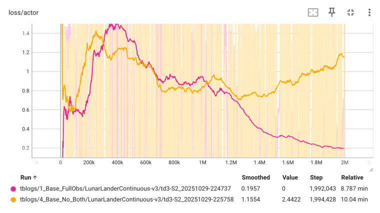
Parameter	**Value / Description**
Environment	`LunarLanderContinuous-v3`
Discount factor γ	0.99999
Replay buffer size	1×10^6 transitions
Batch size	256
Actor hidden layers	[64, 64]
Critic hidden layers	[400, 300]
Learning rate (Actor / Critic)	1×10^{-3}
Optimizer	Adam ($\epsilon = 5 \times 10^{-5}$)
Soft update coefficient τ	0.005
Policy noise σ	0.2 (clipped to ± 0.5)
Policy delay	2
Max epochs	2000 ($\approx$ 2.0M steps)
Memory length k	2 (for observation extension)
Action horizon m	3 (for action extension)

During evaluation, each policy was tested over ten episodes, and performance metrics including actor/critic loss, Q-values, and reward statistics (mean, max, median, min) were recorded.

3.2 Experiment I: Effect of Masking

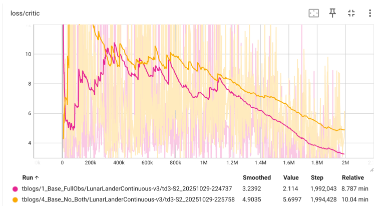
The first experiment investigates the impact of partial observability on the TD3 algorithm by comparing training under fully observable and masked state conditions. Two configurations were tested:

1. **Full Observation (baseline)** — The agent observes the complete state vector $[x, y, v_x, v_y, \theta, \dot{\theta}, c_1, c_2]$.
2. **Masked Observation (partial)** — Certain dynamic variables (e.g., v_x, $\dot{\theta}$) are hidden from the agent, resulting in partially observable inputs.



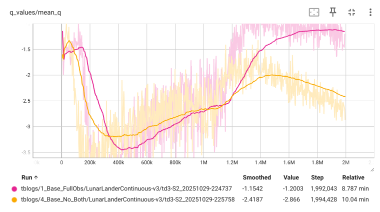
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Base_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-224737	0.1957	0	1,992,043	8.787 min
● tblogs/4_Base_No_Both/LunarLanderContinuous-v3/td3-S2_20251029-225758	1.1554	2.4422	1,994,428	10.04 min

(a) Actor Loss



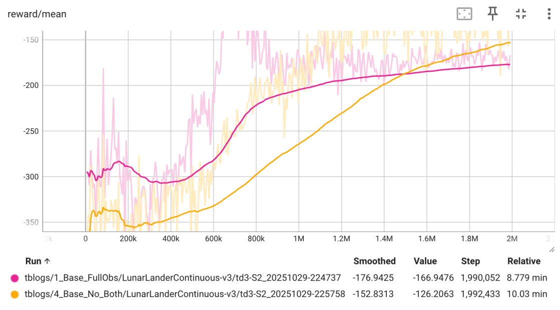
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Base_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-224737	3.2392	2.114	1,992,043	8.787 min
● tblogs/4_Base_No_Both/LunarLanderContinuous-v3/td3-S2_20251029-225758	4.9035	5.6997	1,994,428	10.04 min

(b) Critic Loss



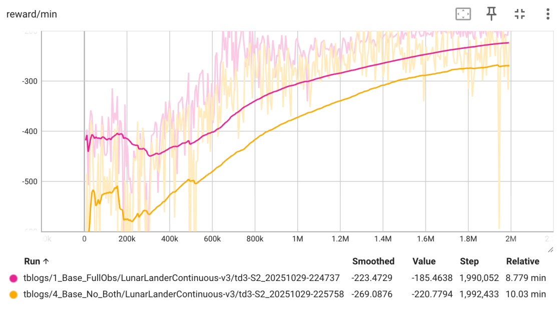
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Base_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-224737	-1.1542	-1.2003	1,992,043	8.787 min
● tblogs/4_Base_No_Both/LunarLanderContinuous-v3/td3-S2_20251029-225758	-2.4187	-2.866	1,994,428	10.04 min

(c) Mean Q-value



Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Base_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-224737	-176.9425	-166.9476	1,990,052	8.779 min
● tblogs/4_Base_No_Both/LunarLanderContinuous-v3/td3-S2_20251029-225758	-152.8313	-126.2063	1,992,433	10.03 min

(d) Mean Reward



Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Base_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-224737	-223.4729	-185.4638	1,990,052	8.779 min
● tblogs/4_Base_No_Both/LunarLanderContinuous-v3/td3-S2_20251029-225758	-269.0876	-220.7794	1,992,433	10.03 min

(e) Min Reward

Figure 1: Problem 1 — Comparison between **FullObs** and **NoBoth** in base TD3.

Training Curve Analysis. The learning curves show distinct differences between the two settings. In the `FullObs` case, both actor and critic losses decrease smoothly after an initial rise and converge to a stable low level. In contrast, the `NoBoth` configuration (with both velocity components masked) displays greater oscillations and slower convergence, indicating less stable parameter updates.

The mean Q-value in the fully observable environment steadily increases and stabilizes around 1.4M steps, whereas the masked version exhibits delayed recovery and large fluctuations in the later phase. Reward trends further highlight this difference: the average reward of the fully observable model improves rapidly after 1M steps, while the masked model lags behind and remains lower overall. The minimum reward curve reveals that the masked configuration maintains a lower bound with persistent volatility, suggesting weaker robustness under extreme states.

Video Observation. The qualitative results are consistent with the quantitative curves. In the `FullObs` video, the lunar lander demonstrates stable thrust control and smooth descent, achieving a controlled soft landing. In contrast, the `NoBoth` model shows erratic motion, frequent angular oscillations, and overcorrections, often resulting in unstable or failed landings.

Summary. This experiment demonstrates that removing key velocity information increases learning difficulty and destabilizes convergence. When the agent cannot observe horizontal and angular velocities, TD3 struggles to estimate motion dynamics accurately, leading to slower convergence, higher critic loss, and overall reduced reward performance.

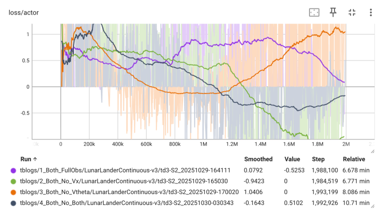
3.3 Experiment II: Performance of Temporal Extension under Partial Observability

This experiment aims to evaluate the effectiveness of the Temporal Extension mechanism in two key aspects. **General Improvement**: whether temporal extension enhances TD3 performance even under fully observable conditions;**Masked Compensation**:whether temporal extension mitigates performance degradation caused by missing state information.

The experiment adopts the Both configuration (combining observation and action extensions) and evaluates four observability settings:

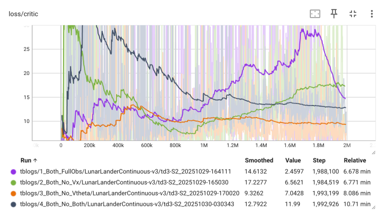
1. **FullObs (fully observable):** all state variables are available, used to assess the general benefit of the extension mechanism.
2. **NoVx:** the horizontal velocity v_x is masked.
3. **NoVtheta:** the angular velocity $\dot{\theta}$ is masked.
4. **NoBoth:** both v_x and $\dot{\theta}$ are masked.

Each configuration was trained for 2.0M steps, with actor/critic losses, Q-values, and reward metrics recorded to evaluate performance differences across observability levels.



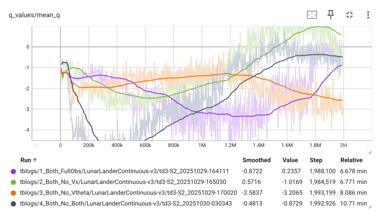
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Both_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-164111	0.0792	-0.5253	1,988,100	6.678 min
● tblogs/2_Both_No_Vx/LunarLanderContinuous-v3/td3-S2_20251029-165030	-0.9423	0	1,984,519	6.771 min
● tblogs/3_Both_No_Vtheta/LunarLanderContinuous-v3/td3-S2_20251029-170020	1.0406	0	1,993,199	8.086 min
● tblogs/4_Both_No_Both/LunarLanderContinuous-v3/td3-S2_20251030-030343	-0.1643	0.5102	1,992,926	10.71 min

(a) Actor Loss



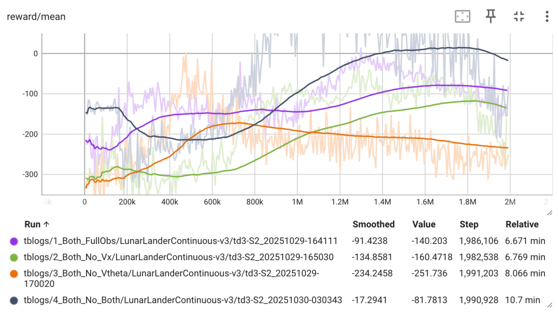
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Both_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-164111	14.6132	2.4597	1,988,100	6.678 min
● tblogs/2_Both_No_Vx/LunarLanderContinuous-v3/td3-S2_20251029-165030	17.2277	6.5621	1,984,519	6.771 min
● tblogs/3_Both_No_Vtheta/LunarLanderContinuous-v3/td3-S2_20251029-170020	9.3262	7.0428	1,993,199	8.086 min
● tblogs/4_Both_No_Both/LunarLanderContinuous-v3/td3-S2_20251030-030343	12.7922	11.99	1,992,926	10.71 min

(b) Critic Loss



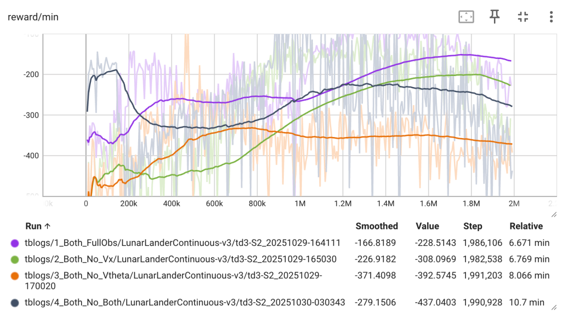
Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Both_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-164111	-0.8722	0.2357	1,988,100	6.678 min
● tblogs/2_Both_No_Vx/LunarLanderContinuous-v3/td3-S2_20251029-165030	0.5716	-1.0169	1,984,519	6.771 min
● tblogs/3_Both_No_Vtheta/LunarLanderContinuous-v3/td3-S2_20251029-170020	-2.5837	-3.2065	1,993,199	8.086 min
● tblogs/4_Both_No_Both/LunarLanderContinuous-v3/td3-S2_20251030-030343	-0.4813	-0.8729	1,992,926	10.71 min

(c) Mean Q-value



Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Both_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-164111	-91.4238	-140.203	1,986,106	6.671 min
● tblogs/2_Both_No_Vx/LunarLanderContinuous-v3/td3-S2_20251029-165030	-134.8581	-160.4718	1,982,538	6.769 min
● tblogs/3_Both_No_Vtheta/LunarLanderContinuous-v3/td3-S2_20251029-170020	-234.2458	-251.736	1,991,203	8.066 min
● tblogs/4_Both_No_Both/LunarLanderContinuous-v3/td3-S2_20251030-030343	-17.2941	-81.7813	1,990,928	10.7 min

(d) Mean Reward



Run ↑	Smoothed	Value	Step	Relative
● tblogs/1_Both_FullObs/LunarLanderContinuous-v3/td3-S2_20251029-164111	-166.8189	-228.5143	1,986,106	6.671 min
● tblogs/2_Both_No_Vx/LunarLanderContinuous-v3/td3-S2_20251029-165030	-226.9182	-308.0969	1,982,538	6.769 min
● tblogs/3_Both_No_Vtheta/LunarLanderContinuous-v3/td3-S2_20251029-170020	-371.4098	-392.5745	1,991,203	8.066 min
● tblogs/4_Both_No_Both/LunarLanderContinuous-v3/td3-S2_20251030-030343	-279.1506	-437.0403	1,990,928	10.7 min

(e) Min Reward

Figure 2: Problem 2 — Comparison among **FullObs**, **NoVx**, **NoVtheta**, and **NoBoth** with Temporal Extension.

Training Curve Analysis. Under the **FullObs** condition, the temporal extension mechanism significantly improves convergence speed and reward level, demonstrating its effectiveness even without any masking. Both actor and critic losses decrease smoothly, Q-values stabilize around 1.4M steps, and average rewards increase consistently, indicating that temporal dependencies improve learning stability and efficiency.

When key velocity information is masked (**NoVx**, **NoVtheta**, **NoBoth**), the overall performance drops, showing slower convergence and larger fluctuations. Nevertheless, the temporal extension provides noticeable compensation: in the **NoVx** case, the model gradually recovers horizontal motion estimation; in **NoVtheta**, attitude control becomes smoother over time; and in **NoBoth**, though learning is the slowest, the model maintains a lower bound and continues improving, revealing enhanced robustness.

Video Observation. The qualitative results are consistent with the quantitative curves. The **FullObs** model achieves stable thrust control and smooth descent, confirming the general improvement of the temporal extension. The **NoVx** model occasionally drifts horizontally but compensates through temporal memory; the **NoVtheta** model shows minor angular oscillations; and the **NoBoth** model starts unstable but later achieves relatively smooth control through historical information and multi-step action prediction.

Summary.

- **The temporal extension improves TD3's convergence and control performance even under full observability.**
- **Under partial observability, it effectively compensates for missing dynamic information, enhancing robustness.**
- **The combined configuration (Both) achieves the best overall performance, demonstrating both generality and adaptability.**

4 Discussion

This study systematically analyzed the performance of the TD3 algorithm under conditions of partial observability and evaluated the improvement brought by the Temporal Extension mechanism in continuous control tasks.

First, **the experimental results show that partial observability significantly weakens the learning stability and control performance of TD3**. When horizontal velocity (v_x) or angular velocity ($\dot{\theta}$) is masked, the agent struggles to accurately estimate the dynamics of the environment, leading to slower convergence, larger loss oscillations, and a lower reward bound. In particular, when both velocity components are hidden (`NoBoth`), training becomes notably unstable, and the policy fails to converge consistently.

Second, **the temporal extension mechanism demonstrates clear performance improvements across all observability settings**. By introducing short-term memory in observations (observation stacking) and short-horizon prediction in actions (action chunking), the agent effectively compensates for missing temporal information. As a result, convergence accelerates and final performance improves even under full observability, while under partial observability, the mechanism helps recover hidden dynamic features, leading to smoother training and more robust control.

However, the experiments also reveal a deeper issue: **although the agent learns precise balance control over attitude and acceleration, it fails to develop global spatial awareness and correction capability**. In several cases, the lander maintains stable thrust control and smooth descent, yet due to the lack of accurate perception of horizontal velocity, it achieves force equilibrium ($\sum F = 0$) while continuing to drift with a constant horizontal velocity, ultimately landing outside the target area. This "stable but misplaced" outcome suggests that the temporal extension primarily improves *local dynamic balance* but does not address the problem of *global spatial controllability*. From a physical standpoint, force equilibrium does not imply zero velocity. Without the ability to estimate and correct long-term positional errors, the agent can maintain posture stability but cannot adjust its trajectory accordingly. This "unaware steady drift" highlights a coupling gap between perception and control layers — the current policy maintains short-term dynamic stability but lacks the global feedback mechanism required to achieve the transition from "attitude stability" to "goal reachability."

In summary, the temporal extension mechanism provides a structurally simple and computationally efficient way to improve TD3's performance under partial observability. It enhances learning stability and local control capability; however, to achieve full spatial controllability and precise trajectory correction, future work should incorporate global position estimation or perceptual modeling, bridging the gap between dynamic balance and spatial correction, and moving toward agents capable of not only balancing but also globally self-correcting motion.